Fast Encoding for Personalized Views Extracted from Beyond High Definition Content

Niels Van Kets, Johan De Praeter, Glenn Van Wallendael, Jan De Cock, and Rik Van de Walle

In: IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 2015.

Fast Encoding for Personalized Views Extracted from Beyond High Definition Content

Niels Van Kets, Johan De Praeter, Glenn Van Wallendael, Jan De Cock and Rik Van de Walle
Multimedia Lab, Ghent University - iMinds, Ghent, Belgium
Gaston Crommenlaan 8 box 201, 9050 Ledeberg-Ghent, BELGIUM

Email: [niels.vankets, johan.depraeter, glenn.vanwallendael, jan.decock, rik.vandewalle]@ugent.be
Telephone: +32 9 33 14957

***Abstract*—Broadcast providers are looking for new opportunities to increase user experience and user interaction on their content. Their main goal is to attract and preserve viewer attention to create a big and stable audience. This could be achieved with a second screen application that lets the users select their own viewpoint in an extremely high resolution video to direct their own first screen. By allowing the users to create their own personalized video stream, they become involved with the content creation itself. However, encoding a personalized view for each user is computationally complex. This paper describes a machine learning approach to speed up the encoding of each personal view. Simulation results of zoom, pan and tilt scenarios show bit rate increases between 2% and 9% for complexity reductions between 69% and 79% compared to full encoding.**

***Index Terms*—Future technologies and services of broadcasting, video coding and processing, High Efficiency Video Coding (HEVC), machine learning, video interaction**

I. Introduction

During the last decade, the broadcasting industry experienced a dramatic increase in spatial resolution for image and video capturing. Nowadays, well known television and movie productions already benefit from native high resolution capturing. At this moment, 4K ultra high definition capturing at a corresponding resolution of 3840 by 2160 pixels is no longer an exception and professional cameras with sensors up to 6K resolutions are for sale. These resolutions can capture a much wider area while preserving the details. If two or more of these cameras are combined to create one very large image, e.g. two 4K images that generate a combined resolution of 7680 by 2160 pixels, it is possible to capture an entire soccer field with the same level of detail as regular sport broadcasts.

Delivering these beyond HD resolutions to a home environment poses some challenges. First of all, current consumer devices support spatial resolutions up to 4K. However, the majority of installed devices are designed for full HD (1920 by 1080 pixels). This means that the majority of consumer home devices are not yet capable of displaying beyond HD content without scaling down or cutting out a region of interest. Second, even if consumer devices were able to display these ultra high resolutions with an acceptable quality and level of detail, the current bandwidths to the home are not high enough to cope with this image size. Even the use of the High Efficiency Video Coding (HEVC) standard [1], which compresses a video with the same quality as its predecessor at half the bitrate, is insufficient to satisfy bandwidth constraints for beyond HD video.

This paper describes a system that allows users to select a subset of the original beyond HD video, such that it can be transmitted to and consumed on regular consumer devices. The goal is to allow each user to select exactly the spatial resolution, place in the field, and zoom level he wants. Consequently, the user can direct his personalized video stream. Since each user might choose a different viewpoint, and encoding typically is a big cost, special effort must be put into reducing the complexity of these simultaneous encodings. This paper describes a solution to reuse encoding decisions from the beyond HD video while keeping the perceptual quality as high as possible. This will reduce the encoding complexity significantly.

The outline of this paper is as follows. Section II contains a brief overview of the HEVC compression standard which is used to compress beyond HD video in this paper. Section III then describes the envisioned architecture together with its main components. In section IV, the extraction and fast encoding techniques of the views are shown. Section V describes the results while the overall conclusion can be found in Section VI.

II. High Efficiency Video Coding

HEVC is a new video compression standard offering better compression efficiency compared to its predecessors [2]. To achieve this, HEVC uses a more flexible scheme for dividing the picture into blocks. The picture is first divided into CTUs of typically 64×64 pixels, which are recursively split into smaller CUs according to a quadtree structure, with the smallest possible block size being 8×8 pixels (illustrated in Fig. 1). Each CU has an associated prediction mode (intra or inter) and is split into one of the eight Prediction Unit (PU) partitioning sizes. Depending on the mode of the CU, each PU contains either intra- or inter- prediction information. For residual coding, each CU is also recursively split into Transform Units (TUs) according to a quadtree structure, with the smallest TU size being 4×4 pixels. Determining the optimal block structure results in the increased complexity of HEVC compared to its predecessors.

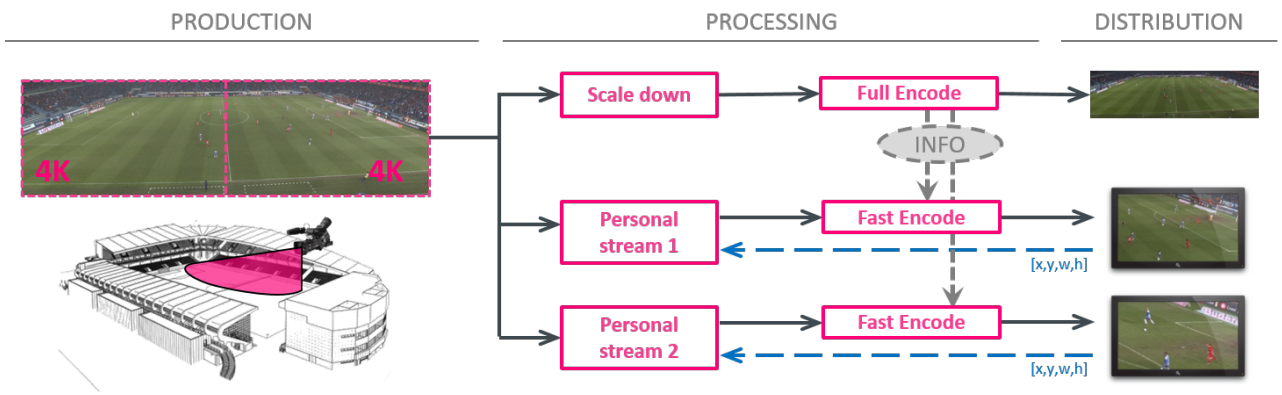

Fig. 2. System architecture of a multi viewpoint fast encoding solution.

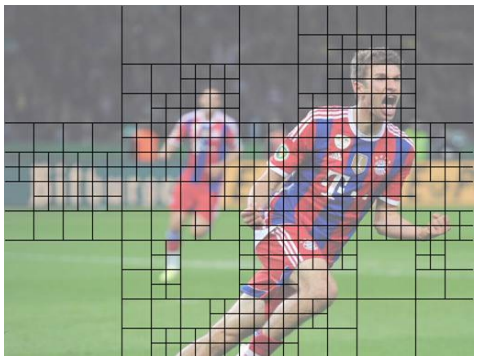

Fig. 1. A possible block division for HEVC video frames.

III. System Architecture

In this section, an overall system architecture that incorporates the creation, processing and distribution of the personalized views is described and depicted in Fig. 2.

The main goal of this system is to deliver a high amount of simultaneously encoded videos with different output resolutions, zoom levels and cropping parameters. In the scenario shown in Fig. 2, multiple users can request different personalized views from within the same ultra high resolution video. The system delivers multiple HEVC streams encoded by fast encoders. These fast HEVC encoders achieve their complexity reduction by reusing coding information from one full HEVC encoder that encodes the source video as a whole. The overall system consists of three major parts: production, processing and distribution. The main focus and contribution of this paper lies within the processing step.

A. Production

This part of the system describes the creation of the input content. In Fig. 2, the content consists of two stitched 4K camera images. This results in content of about 7680 by 2160 pixels. The creation and stitching [3] of this content goes beyond the scope of this paper. In general, any content creation process that results in a very high resolution image can be used as input to this system. Applications may range from sports as depicted in Fig. 2, to surveillance and more industrial applications. The output of this step is directly linked to the processing step.

B. Processing

By using coordinates provided by different users as input to the system, the processing step creates different views and encodes them as HEVC bitstreams ready for distribution to the home. Because a full encode of each view would result in a computationally complex system, this paper describes techniques to reduce this overall complexity. The key idea behind these complexity reducing techniques is the reuse of coding information extracted from a full encode of the beyond HD video to speed up the encoding of a personalized view within the same content.

C. Distribution

The last step in the process is the video distribution. This is the part where the HEVC encoded bitstream is sent to the consumer devices. Because each user gets his own personalized view, the video streams can be optimized based upon the device parameters, e.g. tablet users might request content with a resolution lower than full HD, while users with full HD televisions will want exactly that resolution.

IV. Extraction and Encoding of Views

HEVC uses a block structure as described in Section II to divide a frame into multiple small coding blocks. Since determining the optimal block structure of a frame is computationally complex, the encoder complexity can be greatly reduced by limiting the structures that should be considered. This can be done by exploiting coding information from the encoding of the original Beyond HD video.

A. Alignment and misalignment of blocks

If there is a need to reuse information from one encoder in another encoder that encodes only a subset of the image, two main scenarios can occur. In a first scenario, as depicted in Fig. 3 (a), the view that the user selects is perfectly aligned with existing blocks in the original image. In this case, the block structure of that part of the image can easily be reused [4]. In

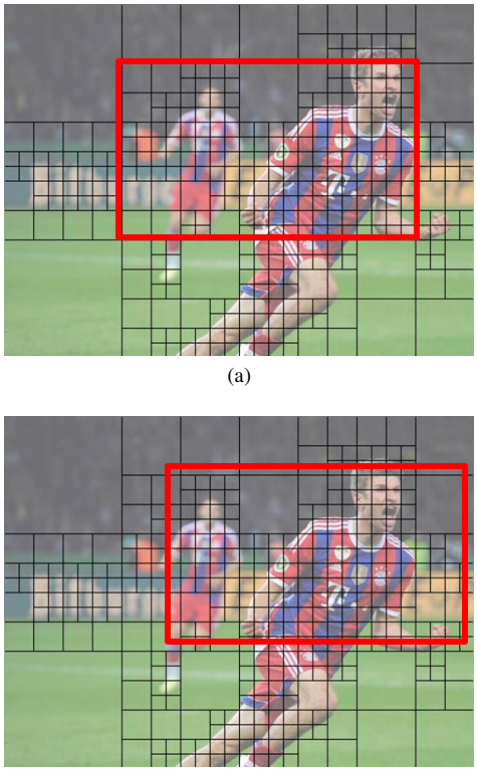

Fig. 3. Difference between spatial alignment (a) and spatial misalignment (b) of selected views.

another case, the user selects a view in which misalignment occurs. In this case, as shown in Fig. 3 (b), the boundaries of the new view do not align with existing block boundaries. This makes reuse of block information less trivial.

B. Virtual pan, tilt and zoom operations

In this paper, three major movements within the video are defined. These movements are similar to the pan, tilt and zoom camera movement known from the field of image capturing. The first movement is panning within the content. As shown in Fig. 4 (a), the pan operation retains the output size, but shifts the image left or right on the horizontal axis. The second movement depicted in Fig. 4 (b) corresponds with tilting. Tilting is equal to panning, only now the motion is upwards or downwards on the vertical axis. The third and last movement is zooming into the content. As depicted in Fig. 4 (c), the zoom operation creates a smaller image by cropping and shifting the original image while retaining the same aspect ratio.

Pan, tilt and zoom operations correspond to a combination of cropping, scaling and shifting the original picture to create a new viewpoint. The outcome of these operations can result in either alignment or misalignment as described in the previous subsection.

C. Exploiting correlation between blocks

Because each view is created from the same input video, there is a certain amount of correlation between the personalized view and the original video. The methodology described in this paper exploits this correlation to reduce the overall encoding complexity.

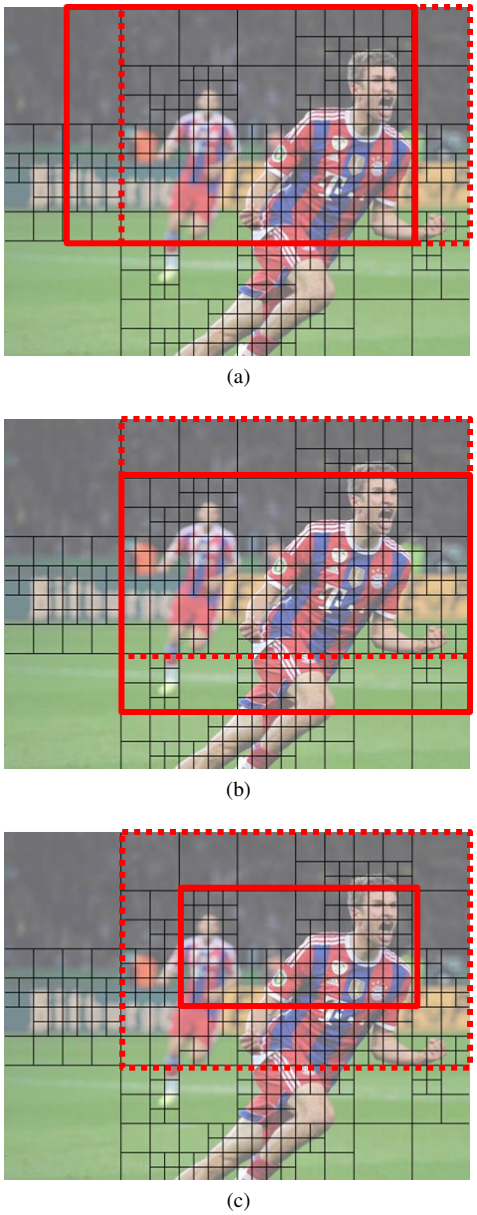

Fig. 4. View adaptation conform to camera pan (a), tilt (b) and zoom (c).

This paper focusses on the HEVC video codec. As described in Section II, HEVC compresses video using flexible block structures. The partitioning of CTUs into CUs and CUs into PUs and TUs is a very costly operation and results in the high computational complexity of HEVC. However, the complexity of encoding multiple views simultaneously can be reduced significantly if (parts of) this block structure information gets automatically predicted from the encoding of the original beyond HD video. In particular, this paper proposes a method to predict the partitioning of CTUs into CUs.

To predict CU structures, a machine learning model is

TABLE I
FULL LIST OF CANDIDATE FEATURES.

Feature index	Description
1	intra fraction
2–5	mean, variance, max, min CU depth
6–9	mean, variance, max, min PU depth
10–13	mean, variance, max, min TU depth
14	motion vector variance
15–16	mean and variance of transform coefficient variance

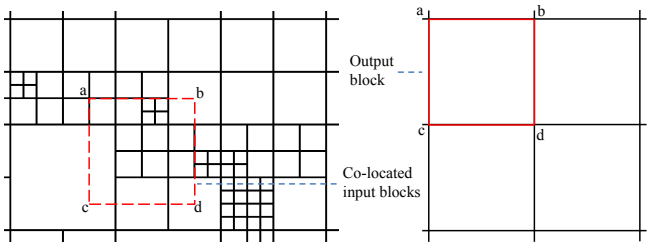

Fig. 5. Example of the co-located blocks in an input sequence (left) for shifting the picture 48 pixels in x- and y-direction (right). If an input block is only partially co-located with the output block, the features associated with the input block need to be weighted.

trained on the first 10 frames of each new view by using the Random Forest algorithm [5]. This machine learning algorithm creates an ensemble classifier from N decision trees. Each tree only uses a random subset of all input features calculated on the coding information of the beyond HD video. Based on this subset, the tree is built by determining rules in each node in order to maximize the entropy reduction for the given samples [6]. In this paper, a node is not split any further if a split would result in a node containing less than 1% of the total number of samples used in the tree.

To determine the output label of a new input sample, each tree in the forest returns the probability of a split for the given samples. These probabilities are then averaged over the complete forest. If the resulting average probability is greater than 50%, the CU is split.

D. Extraction of coding information

In this paper, sixteen candidate features (Table I) are calculated for an output block based on the coding information of the co-located blocks in the input bitstream (Fig. 5) [7]. However, it is possible that only part of an input block is co-located with the output block. Therefore, since information such as the CU depth is determined for a complete block, the mean and variance of this information need to be weighted. For example, if only half of an input block is co-located with the output block, the information will only carry half the weight of a block that is completely co-located with the output block.

The first of the sixteen features is the intra fraction. This value indicates the percentage of pixels in the co-located input blocks that are intra-coded.

Twelve more features are based on the depth of the block structures in the input bitstream. The depth of a block refers to the number of times a Coding Tree Unit (CTU) has been split. Therefore, for Coding Units (CUs) and Transform Units (TUs), depth 0 refers to a block of 64×64 pixels, whereas depth 4 refers to a block of 4×4 pixels. For Prediction Units (PUs), the depth d_{PU} is defined as

$$d_{PU} = \begin{cases} d_{CU} & \text{if } PU_{part\_size} = 2N\times2N \\ d_{CU}+1 & \text{other} \end{cases} \quad (1)$$

with d_{CU} being the depth of the CU to which the PUs belong, PU_{part_size} being the partitioning size of the PUs, and $2N\times2N$ being one of the eight possible partitioning sizes. For each block type (CU, PU, and TU), four features (mean, variance, maximum, minimum value) are calculated for the depths of the co-located input blocks. The use of these features is based on the assumption that the CU structure of a spatially shifted picture will show high correlations with the original block structures.

Another feature is the motion vector variance, which is defined as

$$\sigma_{mv}^2 = \sigma_x^2 + \sigma_y^2 \quad (2)$$

with σ_x^2 and σ_y^2 respectively being the variance of the x and y component of the motion vectors in the co-located input blocks. This feature is used to measure the similarity between the motion vectors of the co-located blocks, since it is assumed that for a small σ_{mv}^2, a good match could be found for the current output block size, meaning that it will not need to be split. If motion vector variance information is unavailable (as is the case with intra-coded blocks), it is not used.

The last two features are the variance σ_{DCT}^2 and the mean μ_{DCT} of the transform coefficient variance, defined as

$$\sigma_{DCT}^2 = 16\sigma_y^2 + \sigma_u^2 + \sigma_v^2 \quad (3)$$

and

$$\mu_{DCT} = \frac{4\mu_y + \mu_u + \mu_v}{6} \quad (4)$$

with σ_i^2 and μ_i respectively the variance and mean of the i-component of the transform coefficient variance. These features are used since the transform coefficient variance will be zero if a block in the input bitstream is skipped. This information might be useful to combine with motion vector variance to predict splitting behaviour of the output block.

V. RESULTS

To evaluate the algorithm for fast encoding of personalized views, version 16.3 of the HEVC reference software was modified [8]. The Ultra HD (UHD) sequences *Bosphorus*, *Bund-Nightscape*, *Jockey*, *Marathon*, *ParkJoy* and *ReadySteadyGo*, which have a resolution of 3840×2160 pixels, were used as the original video from which personalized views are extracted. These sequences are further described in Figure 6 and Table II. These sequences were chosen from a larger pool of UHD sequences due to their diverse spatial and temporal activity as measured in Spatial Index (SI) and Temporal Index (TI) (Fig. 7) [9].

(a) Bosphorus (b) BundNightscape (c) Jockey

(d) Marathon (e) ParkJoy (f) ReadySteadyGo

Fig. 6. Ultra HD sequences used in the simulations.

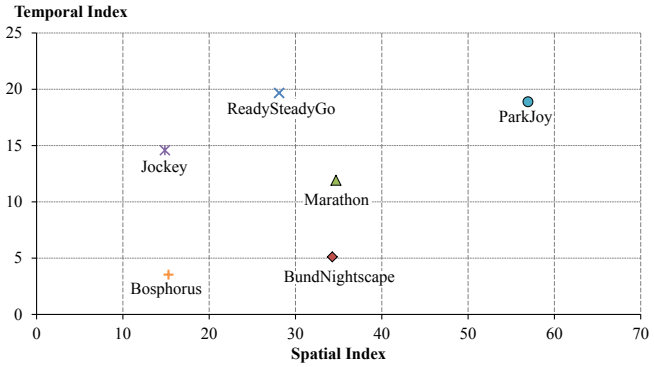

Fig. 7. Description of the chosen test sequences in terms of spatial and temporal activity.

TABLE II
SHORT DESCRIPTION OF TEST SEQUENCES, INCLUDING THE FRAME RATE IN FRAMES PER SECOND (FPS). THE TOTAL NUMBER OF ENCODED FRAMES EQUALS TEN TIMES THE FRAME RATE WITH A MAXIMUM OF 600 FRAMES.

Sequence	Fps	Description
Bosphorus	120	Camera moves in parallel to follow boat
BundNightscape	30	Fixed camera and small moving objects
Jockey	120	Fast camera pan and zoom to follow horse
Marathon	30	Fixed camera and many moving objects
ParkJoy	50	Camera moves in parallel to follow people
ReadySteadyGo	120	Camera pans to follow action

Each UHD sequence was encoded with quantization parameters (QP) 22, 27, 32 and 37. Each of these versions was encoded using a low delay configuration, which consists of an I-frame followed by P-frames. This configuration results in a lower delay, which is a requirement for interacting with personalized views.

To evaluate the proposed algorithm, the difference in compression efficiency and encoding complexity reduction are measured. The difference in compression efficiency is expressed in Bjøntegaard Delta (BD) rate [10]. This metric shows the average increase in bitrate for the same Peak Signal-to-Noise Ration (PSNR) of encoding a personalized view by reusing information from the original UHD sequence (fast encoder) compared to encoding this view without reusing information (reference encoder).

Complexity reduction is determined by comparing the encoding time of the fast encoder T_{fast} to the encoding time of the reference encoder T_{ref} in terms of time saving (TS):

$$TS(\%) = \frac{T_{ref} - T_{fast}}{T_{ref}} \tag{5}$$

In the following subsections, two scenarios are simulated and evaluated. First, a virtual zoom is simulated, followed by a pan- and tilt-scenario.

A. *Simulating zoom*

If a user with a 720p screen wants to view a part of the UHD video at the original resolution, the CU structure of the relevant part of the UHD video can be copied to encode this personalized view. However, if the user zooms out with a factor of two, the personalized view should reuse the information from a 1080p version of the original video. If the user zooms out even further to watch the complete video on the 720p screen, CU structure information for a 720p version needs to be determined. To simulate this scenario of zooming, the UHD video is encoded at three resolutions (2160p, 1080p and 720p) by using coding information of the UHD video encoded at 2160p.

The results in Table III indicate that performing a fast encode at resolutions other than 2160p does not differ greatly in compression efficiency (with the exception of *Jockey*) from

TABLE III
RESULTS OF THE ZOOM SIMULATION.

	BD-Rate(%)			Time Saving (%)		
Sequence	2160p	1080p	720p	2160p	1080p	720p
Bosphorus	4.3	4.6	5.1	78.8	77.8	77.4
BundNightscape	4.2	5.5	5.5	79.5	77.9	78.2
Jockey	4.7	8.2	9.3	78.0	77.6	77.5
Marathon	3.2	3.6	2.9	71.0	69.2	69.4
ParkJoy	2.4	2.5	2.1	72.1	69.2	69.2
ReadySteadyGo	4.5	5.6	5.5	77.1	74.3	73.8

performing a fast encode of the UHD sequence at its original resolution of 2160p. This seems to indicate that zoom levels of 1080p and 720p are indeed viable choices for providing personalized views. Additionally, all versions also show similar complexity reductions between 69% and 79%.

If the CU structure of encoding the UHD sequence at its original resolution is predicted, the machine learning algorithm reports a 100% accuracy since the encoder simply has to copy all of the original CU information. However, the simulation reported BD-rates between 2.4% and 4.7% (Table III), which is far from the expected 0%. This behavior was investigated further and can be attributed to two encoder optimizations in the HM reference software.

The first encoder optimization speeds up inter-prediction of asymmetrical motion partitions [11]. This optimization uses the PU partitioning size of the parent CU to decide which asymmetrical motion partitions will be evaluated and whether full motion estimation will be performed in addition to merge estimation. If the evaluation of the parent CU is skipped, as is the case with the algorithm proposed in this paper, its PU partitioning size is not known and the existing encoder optimization does not function correctly.

The second encoder optimization provides an extra candidate starting point for motion estimation [12]. This starting point is based on the motion vector of the most recently calculated $2N\times2N$ PU with the same reference picture as the currently tested motion vector. However, if some CU sizes are not evaluated, this value can be incorrect, leading to less optimal encoder decisions.

Since the above encoder optimizations appear to conflict with the proposed algorithm, future work will investigate equivalent, non-conflicting optimizations.

B. Simulating pan and tilt

The effects of pans and tilts of the personalized view can be simulated by shifting the view on the UHD video in x- and y-direction. Since the size of CTUs in this paper is 64×64 pixels, shifts of k pixels are assumed to be equivalent to shifts of $k \mod 64$ pixels in terms of misalignment. Additionally, a shift of k pixels is also assumed to be equivalent to a shift of $-k$ pixels. Hence, only shifts of up to 32 pixels will be evaluated. To simulate these shifts, a view of 3808×2128 pixels is used on the UHD sequences. This view can shift up to 32 pixels in both x- and y-direction (see also Fig. 8).

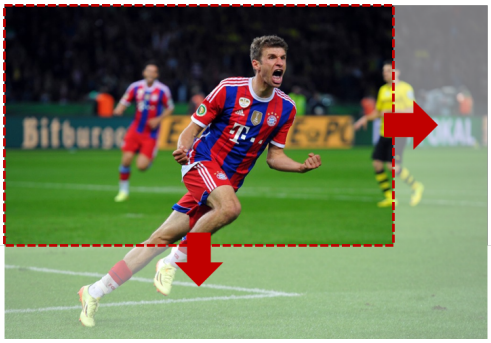

Fig. 8. To simulate pan and tilt for shifts of up to 32 pixels in both x- and y-direction, the personalized view shows the complete video with 32 pixels subtracted from both its width and height.

TABLE IV
EFFECT ON BD-RATE WHEN COMBINING PAN AND TILT MOVEMENTS, FOR THE SEQUENCE BUNDNIGHTSCAPE.

	x-shift				
y-shift	0	8	16	24	32
0	4.2	5.7	5.1	5.7	4.4
8	5.7	6.0	6.0	6.2	5.6
16	5.2	5.9	5.4	6.0	5.1
24	5.8	6.2	6.2	6.2	5.9
32	4.6	5.7	5.2	5.7	4.5

In a first experiment, the relation between shifts in different directions was investigated. As seen in Table IV for the sequence *BundNightscape*, an (x, y) shift shows a similar compression efficiency as a (y, x) shift. For this sequence, the largest difference can be seen between e.g. a shift of (0,32) pixels with a BD-rate of 4.6% and (32,0) pixels with a BD-rate of 4.4%. Since the other sequences displayed similar behavior, only shifts in a single direction are used in the next experiment.

In a second experiment, the effect of misalignment was investigated in detail for all 32 shifts. As seen in Fig. 9, some shifts perform better than others. Shifts of 0 and 32 pixels perform best since they respectively preserve alignment with the CTU-grid of 64×64 pixels and a CU-grid of 32×32 pixels. Shifts of 16 pixels perform slightly better than surrounding shifts, although this performance is worse than for shifts of 32 pixels. Depending on the sequence, shifts from 1 to 3 pixels and shifts of 30 and 31 pixels also perform generally better than other shifts. This is most likely due to very small shifts only introducing a negligible amount of misalignment for some sequences.

Jockey and *Bosphorus* behave atypical in a sense that the BD-rate does not change much between different shifts. As seen in Fig. 7, both sequences have a lower spatial activity. The relation between this behavior and the spatial activity should be further investigated.

In all of the above results, the fast encoder predicts the

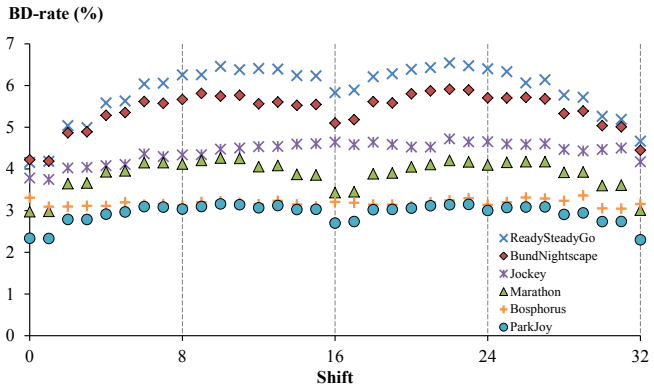

Fig. 9. Effect on BD-rate when panning the virtual camera.

TABLE V
TIME SAVING WHEN PANNING THE VIRTUAL CAMERA.

	Time Saving (%) (all shifts)	
Sequence	Average	Standard deviation
Bosphorus	79.0	0.2
BundNightscape	79.0	0.1
Jockey	77.4	0.1
Marathon	70.0	0.2
ParkJoy	71.2	0.3
ReadySteadyGo	76.1	0.2

complete CU structure. As seen in Table V, this means that the TS is similar for all shifts, since only a single CU structure is evaluated by the encoder. Small variations between shifts may occur if the model predicts large CU sizes more often, since this reduces the number of evaluated CUs. Since the complexity reduction is similar for all shifts, only the compression efficiency should be taken into account to determine the shifts that should be allowed when selecting personalized views.

VI. CONCLUSION

In this paper, a system for encoding personalized views extracted from beyond HD content is presented. To handle the scalability issues of encoding many different views in parallel, a method is proposed to reduce the encoding complexity of each view. This method exploits the correlation with the original encoded beyond HD video and the personalized view.

Simulation results of zoom scenarios show bit rate increases between 2% and 9% with complexity reductions between 69% and 79% compared to full encoding. The results also show that zooming at three different zoom levels does not make a large difference in compression efficiency and complexity reduction.

When panning and tilting the virtual camera, the bit rate increases between 2% and 7% with complexity reductions between 70% and 79%. In this case, the amount of misalignment does have an effect on the compression efficiency. As a result, in a system with personalized views, the view should only be shifted with multiples of 32 pixels compared to the original encoded video.

ACKNOWLEDGMENT

Part of the research leading to this publication was performed in the High Tech Visualisation research program (HiViz) of iMinds. Additionally, the activities described in this paper were funded by Ghent University, iMinds, the Agency for Innovation by Science & Technology (IWT), the Fund for Scientific Research (FWO Flanders), and the European Union, and were carried out using the Stevin Supercomputer Infrastructure at Ghent University.

REFERENCES

[1] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[2] J. Ohm, G. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards – Including High Efficiency Video Coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec 2012.

[3] C. Fehn, C. Weissig, I. Feldmann, M. Muller, P. Eisert, P. Kauff, and H. Bloss, "Creation of High-Resolution Video Panoramas of Sport Events," in *Proc. IEEE Int. Symposium Multimedia (ISM)*, Dec 2006, pp. 291–298.

[4] J. De Praeter, J. De Cock, G. Van Wallendael, S. Van Leuven, P. Lambert, and R. Van de Walle, "Efficient Transcoding for Spatially Misaligned Compositions for HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct 2014, pp. 2494–2498.

[5] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001.

[6] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[7] L. P. Van, J. De Praeter, G. Van Wallendael, J. De Cock, and R. Van de Walle, "Machine learning for arbitrary downsizing of pre-encoded video in HEVC," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan 2015, pp. 406–407.

[8] K. McCann, C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. Sullivan, "High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Improved Encoder Description," ITU-T Joint Collaborative Team on Video Coding (JCT-VC), Tech. Rep. JCTVC-S1002, Oct. 2014.

[9] B. Akoa, E. Simeu, and F. Lebowsky, "Using classification for video quality evaluation," in *Proc. IEEE Int. Conf. Microelectronics (ICM)*, Dec 2013, pp. 1–4.

[10] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T Video Coding Experts Group (VCEG), Tech. Rep. VCEG-M33, Apr. 2001.

[11] I.-K. Kim, W.-J. Han, J. H. Park, and X. Zheng, "CE2: Test results of asymmetric motion partition (AMP)," ITU-T Joint Collaborative Team on Video Coding (JCT-VC), Tech. Rep. JCTVC-F379, July 2011.

[12] B. Li and J. Xu, "On motion estimation start point," ITU-T Joint Collaborative Team on Video Coding (JCT-VC), Tech. Rep. JCTVC-R0105, July 2014.